Advanced Data Analysis

STAT 9850B FINAL PROJECT

"Stock price prediction using machine learning algorithms"

Instructor: Dr. Li-Pang Chen

Group 8

April 2020

Department of Statistical and Actuarial Sciences

Western University

Contents

Abstract

Three different machine learning techniques including Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN) and Support Vector Machine (SVM) are used in computer codes[1][2] to perform a one-day forecast on stock prices. Obtained from Yahoo Finance historical data, four different stocks are selected from different industries with different patterns to evaluate the accuracy of each model while the Adjusted close price of the next day is the response variable. Adjusted close prices of previous days for certain time frames are considered as the inputs of the LSTM and CNN model and it was unveiled that combination of CNN and LSTM improves prediction accuracy. Also, the Close price, Open price, Daily low, Daily high, Adjusted Close price, and Volume of trades of the last 30 days are considered as the inputs of the SVR model and it was seen that the prediction accuracy improves when using these variables.

Keywords: LSTM, CNN, SVR, Stock price prediction.

[1] https://colab.research.google.com/drive/1ujAeQM6f_ZzKByjQALfb-sK60kWvoIXa
[2] https://colab.research.google.com/drive/1pA1gl3m5SP5jEfDGXsplWBF8BO602tKC

1 Introduction

1.1 Background and Purpose of the Project

Investors in the stock markets have been attempting for a long time to develop methods for predicting stock prices to make more profit. It is, however, complicated to make precise and consistently correct forecasts. The reason lies in the fact that there are various factors affecting investors' sentiments and thus, causing stock market movements. These factors could be international events such as a military war or even some news about political leaders' decisions. Considering the complex nature of such factors and all the contributing hyper-parameters, developing a consistent and accurate method is not easy. It is, therefore, essential to construct a proper model using the right dataset to generate reliable results.

A good model considers all the essential variables that drive market prices and, therefore, perform robustly and accurately under the real-life scenarios [1]. One could divide the studies of stock market trends into two general categories: Fundamental approach where earnings, expenses, assets, and liabilities of a company are assessed to predict price trends and the technical procedure where all the factors as mentioned above are assumed to be reflected in the stock price itself. Time series analysis has been traditionally used in technical approaches. The most popular method in modeling time series data has been Auto Regressive Integrated Moving Average (ARIMA) models proposed by Box & Jenkins [2]. However, this is a linear model and does not consider the volatility or the movements in the variance of the underlying stock prices. That makes ARIMA unsuitable for forecasting stock price data, especially when there are fluctuations in the volatilities. However, with the recent advancements in computer and data science, Artificial Neural networks (ANNs) and machine learning are now the most popular methods when it comes to technical analysis and they form the motivation for conducting this research work [3].

Extensive researches have been carried out to exploit machine learning methods for the studies of stock markets. Huang used several different machine learning techniques based on fundamental analysis and showed in his thesis that machine learning could be utilized to assist fundamental analysts in their decision making and strategy planning [4]. Huang et al. used financial time series analysis based on the wavelet kernel support vector to forecast the Nasdaq composite index [5]. They showed that using wavelet kernel SVMs can increase prediction accuracy compared with the polynomial kernel SVM and Gaussian kernel SVM. Wu & Lu studied several individual and hybrid computational intelligence methods based on statistical learning algorithm and evaluated the performance of each model [6]. Kao et al. proposed a method to integrate wavelet transform, Multivariate Adaptive Regression Splines (MARS), and Support Vector Regression (SVR) to improve prediction accuracy [7]. Tipirisetty conducted a textual analysis of online news and blogs using neural networks to investigate their impact on upward or downward movements of the stock market [8].

While most of the previous works are focused on developing and improving a particular machine learning algorithm, this research aims to develop different machine learning methods and compare their performances based on their accuracies. The models used in this work are LSTM, SVR, and CNN. They are then used to predict stock prices of Apple, Mastercard, Ford, and ExxonMobil, to include companies from different industries with different trends. The data have been obtained from Yahoo Finance. The daily variables of each stock are considered as models' features with a lookback window of 30 to 180 days. This approach, of course, creates a high-dimensional dataset that needs to be pre-processed for use in machine learning models. Methodologies for different models, i.e., SVM, LSTM, and CNN are presented in Section 2 of this presentation. In Section 3, a comprehensive data analysis is performed on the datasets, and different models are trained to predict stock prices. Finally, Section 4 presents conclusions for the proposed models.

1.2 Data Description

The datasets used in this project include Apple, Mastercard, Ford, and ExxonMobil historical data obtained from Yahoo Finance. The data includes Open price, Close price, Daily high price, Daily low price, Adjusted close price and the Volume of trades for these stocks from January 1st, 2002 until March 11th, 2020 (except for Mastercard where data is available from 2006). The Adjusted close price of these stocks are shown in the Figure1-1. (Daily log-returns presented in Appendix A).

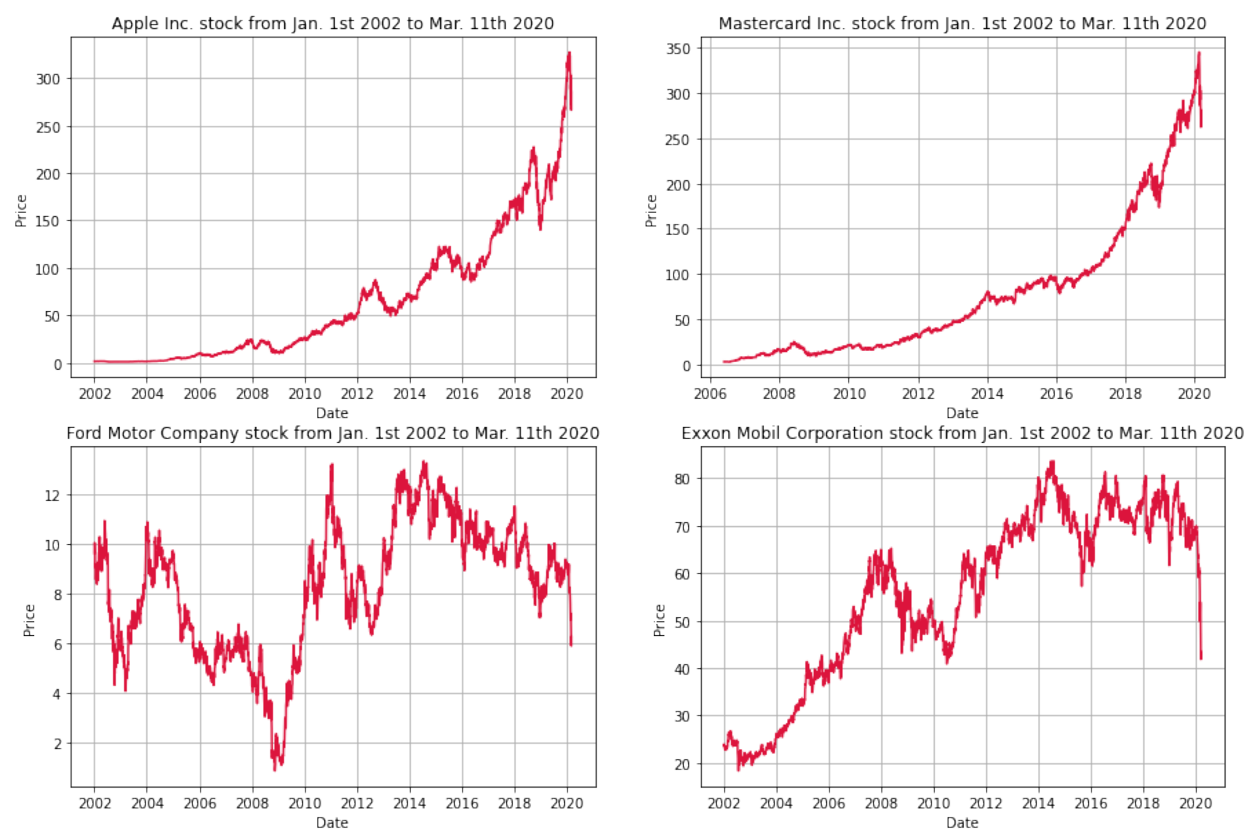

Figure 1-1: Stock prices for Apple, MasterCard, Ford, and ExxonMobil

Traditionally financial time series are assumed to be stationary, and if not, number of techniques can be used to transform data into a stationary process. Other than the stationarity, ergodicity is also required to ensure that the ensemble average for a quantity converges to the expected value of that quantity's time average. Regarding these issues, investigating statistical properties of data is a common approach when it comes to stock price forecasting. A comprehensive analysis was conducted on the properties of datasets and the results are presented in Appendix A. Some of these properties which were investigated for our data are:

- **Sharp peak and heavy tailed (non-normal) distribution of returns (contrary to the Black-Scholes Normal distribution assumption)**
  This can be noticed by referencing to Table 1-1. It is also shown in Figure A-2, Figure A-3 and Figure A-4 that log-returns distributions are better fitted to a Student-t distribution rather than the normal distribution.

Table 1-1: Kurtosis and Skewness metrics of the financial data

	Kurtosis	Skewness
AAPL	5.642884	-0.199756
MAST	8.719550	0.362088
FORD	17.013923	-0.016349
EXON	12.596108	-0.194201

- **The high degree of variability in returns which is observed in the form of many bursts in returns plot (as is shown in Figure A-1)**

- **Volatility clustering:**
  There are periods of high volatility observable in both Figure A-1 and Figure A-7

- **Weak (linear) correlation between the returns and their lagged values (as is shown in Figure A-6)**
  One of the critical characteristics of stock price and return data is the presence or absence of dependence between price increments. The presence of linear dependence can be readily checked by the autocorrelation function. Still, it is well known that in efficient markets, the price follows a random walk, and the linear dependence, if present, will decay rapidly to zero. This is where the traditional time series approaches cannot discriminate returns and white noise (containing all frequencies).

- **Long range dependence detectable by nonlinear autocorrelations of returns**

  As mentioned above, the linear independence fades when dependence is investigated on the squared return data. As shown in Figure A-7, there is considerable dependence especially on the most recent returns. However, this dependence also dissipates as time moves forward.

2 Model and Methodology

2.1 Support Vector Machines (SVM)

Support vector machines (SVM) were initially proposed as a semi-parametric classification method [9], which uses a hyperplane (linear surface) as a boundary between the two classes of observations maximizing the distance or the margin of the nearest point from each class to that separating hyperplane. However, this simple definition of SVM is valid only when the classes are separable by a linear boundary. If the classes are not separable, the margins are relaxed by soft margins, which allows for some observations to be within the boundary or even to the other side of the hyperplane. SVM is semi-parametric in the sense that it assumes a linear boundary, which is a parametric surface; however, it does not assume any probability models for the observations like Linear Discriminant Analysis (LDA) or Logistic Regression. If the boundary is truly non-linear, then a linear hyperplane based SVM cannot satisfactorily classify observations into classes. To overcome this problem, the observations are projected into higher dimensional space using kernel-based basis functions so that the boundary becomes linear in that higher dimensional space. The choice of tuning parameters in the kernel function provides the flexibility of the SVM models to model the right nonlinear boundary.

To describe the SVM classifier briefly, let us consider a training data $(y_i,\ x_i)$ for $i = 1, \dots, T$ where $y_i = +1$ for upward movement of the stock prices and $y_i = -1$ for downward movement. x_i is the vector of covariate and past stock prices. Then a classifier $f(x) \geq 0$ if $y_i = 1$ and $f(x) < 0$ for $y_i = -1$ will optimally classify the observations, that is, for correct classification, we must have $y_i f(x_i) > 0$. A linear classifier will have the form

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = \beta_0 + \beta^\top x. \tag{1}$$

In support vector classification, we maximize the margin of this hyperplane to the nearest points from the two classes, that is, $y_i f(x_i)$. However, any scalar multiple of β will produce the same hyperplane but will have larger values for $y_i f(x_i)$. Thus, β needs to be normalize so that the points on the margin have $y_i f(x_i) = +1$ and -1 , respectively for the two classes. Let x_+ be a point on the margin on the positive side, and x_- be a point on the negative side. Then the margin is, as:

$$\frac{\beta}{\|\beta\|}(x_+ - x_-) = \frac{\beta^\top x_+ - \beta^\top x_-}{\|\beta\|} = \frac{2}{\|\beta\|}. \tag{2}$$

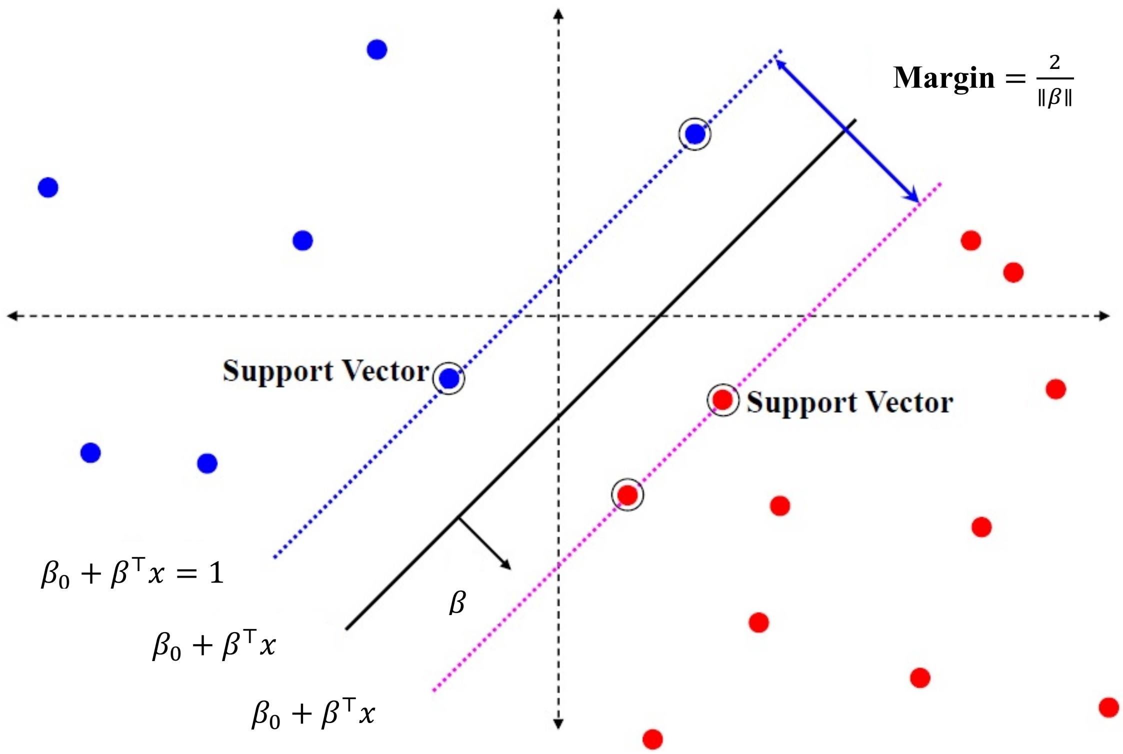

Figure **2-1**: Sketch of a support vector classifier [10]

Therefore, the optimization problem can be written as

$$\text{maximize} \frac{2}{\| \beta \|}$$

Subject to

$$y_i(\beta_0 + \beta^{\top}x) \geq 1 \;\; \text{for } i = 1, \dots, T.$$

Equivalently, one can write

$$\text{minimize} \frac{1}{2} \| \beta \|^2$$

subject to

$$y_i(\beta_0 + \beta^{\top}x) \geq 1 \;\; \text{for } i = 1, \dots, T.$$

This is a quadratic optimization problem with linear constraints.

As mentioned above, the classes may not be entirely separable, and hence, we may require to introduce a soft margin or to allow for some points to be within the margin and/or even misclassified with some error bounds. The optimization problem becomes

$$\text{minimize } \tfrac{1}{2} \| \beta \|^2 + C \textstyle\sum_{i=1}^{T} \epsilon_i \tag{3}$$

subject to

$$y_i(\beta_0 + \beta^\top x) \geq 1 - \epsilon_i, \qquad \epsilon_i > 0, \text{ for } i = 1, \ldots, T.$$

Here C is a regularization parameter with a large value of C, implying a narrow margin and small value of C, implying a large margin. We can write the optimization problem as

$$\text{minimize} \frac{1}{2} \parallel \beta \parallel^2 + C \sum_{i=1}^{T} \max\left(0, 1 - y_i(\beta_0 + \beta^T x_i)\right) \tag{4}$$

which is a convex optimization problem. The dual version of the optimization problem can be written as

$$\max_{\alpha_i \geq 0} \sum_{i=1}^{T} \alpha_i - \frac{1}{2} \sum_{j=1}^{T} \sum_{k=1}^{T} \alpha_j \alpha_k y_j y_k x_j^\top x_k \text{ subject to } 0 \leq \alpha_i \leq C, \sum_{i=1}^{T} \alpha_i y_i = 0. \tag{5}$$

If we consider a dual version of the classifier $f(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i y_i (x_i^T x) + b$ then the objective function of the dual becomes

$$\sum_{i=1}^{T} \alpha_i - \frac{1}{2} \sum_{i=1}^{T} \alpha_i y_i f(x_i). \tag{6}$$

In order to accommodate for the nonlinear boundary, the feature vectors x_i is projected on to a higher dimensional space through a nonlinear function ϕ: $\mathbb{R}^p \to \mathbb{R}^d$, where $d \geq p$ and then apply SVM classifier so that the classifier becomes $f(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i y_i \left(\phi(x_i)^\top \phi(x)\right)$. In general, the inner product between $\phi(x_i)$ and $\phi(x)$, $\phi(x_i)^\top \phi(x)$ can be replaced by the kernel function $k(x_i, x)$ to have the classifier

$$f(x) = \sum_{i=1}^{T} \alpha_i y_i k(x_i, x). \tag{7}$$

For the regression problem, where we are interested in forecasting the stock prices instead of just predicting a downward or upward trend, our response y_i represent the real value. The optimization problem can be changed to

$$\text{maximize } \frac{1}{2} \sum_{i=1}^{T} (\alpha_i - \alpha_i^*)\left(y_i - f(x_i)\right) - \epsilon \sum_{i=1}^{T} (\alpha_i + \alpha_i^*) \tag{8}$$

subject to

$$\sum_{i=1}^{T} (\alpha_i - \alpha_i^*) = 0 \text{ and } 0 \leq \alpha_i \leq C \tag{9}$$

where

$$f(x) = \sum_{i=1}^{T} (\alpha_i - \alpha_i^*) k(x_i, x) + b. \tag{10}$$

Here, the absolute errors smaller than ϵ are disregarded. $C > 0$ is a regularization constant determining a trade-off between non-linearity of f and the amount up to which deviations larger than ϵ can be considered. The most popular choices of the kernel functions are radial basis functions $k(x, y) = \exp\left(-\gamma \parallel x - y \parallel^2\right)$ [11], and polynomial kernel $k(x, y) = (\gamma x^\top y + r)^d$ [12].

2.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory or LSTM is a variation of the recurrent neural network, which avoids long term dependence problem and makes it suitable for predicting financial time series, including stock prices. Hochreiter & Schmidhuber [13] proposed LSTM, which replaced the hidden layer of neurons in RNN with memory cells. The key feature is the state of these memory cells in the hidden layer, and these states are updated with the input through the gate structure, as shown in Figure 2-2. Each of the memory cells contains three layers of sigmoid transformation and one layer of tanh transformation.

At period t, the memory cell takes the output of the period $t-1$, h_{t-1}, and the external input or covariates x_t at period t as inputs and uses a sigmoid transformation

$$f_t = \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right) \tag{11}$$

where W_f and b_f are the matrices of weights and the bias vector, respectively, of the forgotten gate. This will result in a value between 0 and 1, where 0 indicates discarding all information and 1 presents keeping all information from the previous period.

The input gate then determines the state of the cell which needs to be updated by the sigmoid function

$$I_t = \sigma\ (W_t . [h_{t-1}, x_t] + b_t). \tag{12}$$

It also updates the information needs to be updated to the cell at period t, using a tanh function

$$\hat{C}_t = \tanh(W_c . [h_{t-1}, x_t] + b_c). \tag{13}$$

Finally, the state of the memory cell is updated using

$$C_t = f_t * C_{t-1} + I_t * \hat{C}_t. \tag{14}$$

The output information is determined by using a sigmoid layer and then processed as a tanh function as follows

$$O_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \tag{15}$$

and the final output from the cell

$$h_t = O_t * \tanh(C_t). \tag{16}$$

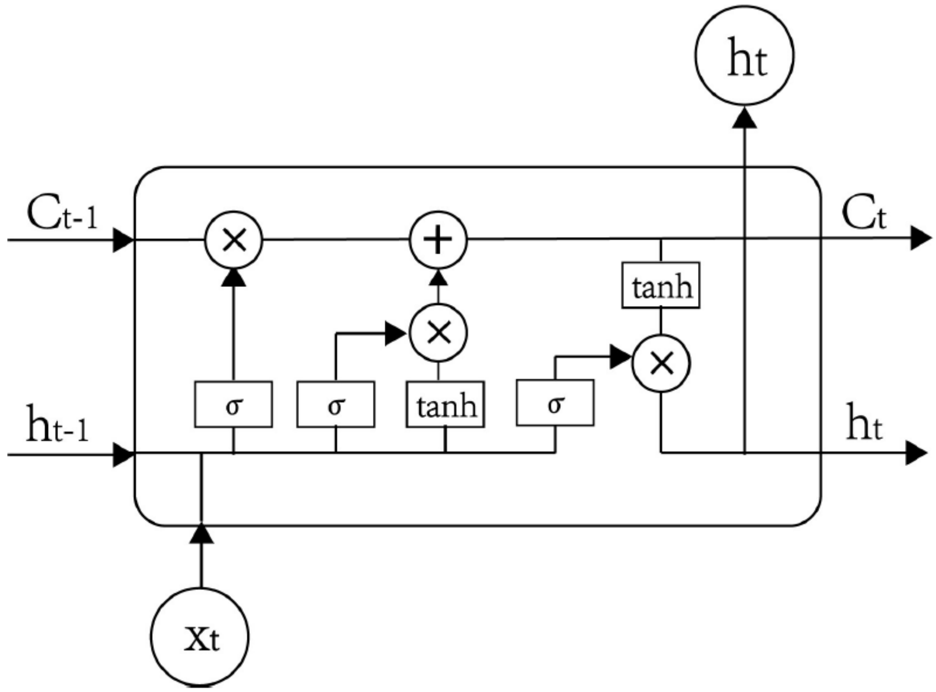

Figure 2-2: The memory cell structure of an LSTM model [14]

2.3 Convolutional Neural Networks (CNN)

Currently, Neural Networks is one of the most popular machine learning algorithms. There are different types of Neural Networks methods such as CNN (Convolutional Neural Networks), Feedforward Neural Network (FNN), and RNN (Recurrent Neural Networks). Traditionally, the most application of CNN is in the image processing. We can utilize the CNN method for prediction of the stock price. In the literature, researches [15, 16, 17] who have applied the CNN method for stock market prediction used a CNN, which took a one-dimensional input for making predictions only based on the history of closing prices while ignoring other possible sources of information like technical indicators.

The CNN contains at least three layers. They include convolution layers, pooling layers (e.g., max pooling), and fully connected (FC) layers. The convolutional layer performs a series of convolutional operations on its inputs to facilitate pattern recognition, and each input is convoluted with kernel or filter. In convolution layers, there are three features that we should determine. They are the length and number of kernels, pooling stride, and padding.

The pooling layers have the responsibility to divide the dimension of the input, which is the most relevant information preserved. To utilize its down sampling function, a pooling window and a pooling stride should be configured. The fully connected layers (FC) combine each neuron to accurately and efficiently classify each input.

3 Notation and Algorithm

This Section describes the notation and algorithm utilized in this study. A computer program has been developed in Python for data analysis, pre-processing of the dataset, and implementing the machine learning algorithms. Following steps provide an overview of how the method is implemented.

1- Four stocks of Apple, Mastercard, Ford, and ExxonMobil are selected. Stock prices for January 1st, 2002 until March 11th, 2020, are downloaded (except for Mastercard data which is available from 2006).

2- Data processing is performed on the dataset to study the distribution of prices and daily log-returns, the correlations between different stocks, and the autocorrelations in historical prices.

3- LSTM is applied to the Apple dataset considering a lookback window of 180 days on Adjusted close prices as the response variable. Figure 3.1 displays the architecture developed as a sequential model using LSTM and Dense layers. This would create a 180-feature dataset. The model is then trained on 75% of each dataset with validation split and batch size being 0.25 and 256, respectively. ADAM is used as the optimizer with a learning rate of 0.001. Training loss and validation loss are monitored, and it is observed that convergence is achieved after 100 epochs.

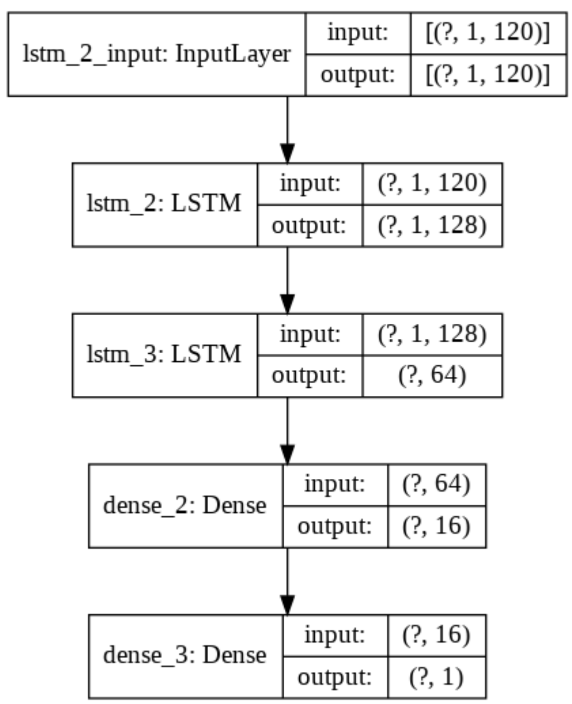

Figure 3-1: Developed architecture as a sequential model using LSTM and Dense layers

4- Steps 3 and 4 are carried out for the other three stocks.

5- Another model on the entire Apple dataset is trained using LSTM and is then used to predict the different three stock prices. This is to investigate whether it is possible to use a model trained on one stock to make forecasts on another one.

6- The same approach in steps 3, 4, 5, and 6 is repeated this time for a machine learning model using both CNN and LSTM (a combination of CNN and LSTM). Figure 3.2 displays that architecture developed as a sequential model using a Convolutional layer, dense layers, CNN, and LSTM. The inputs of the model are a 120-day lookback window of Adjusted close prices. This would create a 120-feature dataset. The model is then trained on 75% of each dataset with kernel size of 8, strides equal to 1, and the activation function being ReLU. ADAM is used as the optimizer with the learning rate of 0.001. Training and validation losses are monitored, and it is observed that convergence is achieved after 100 epochs.

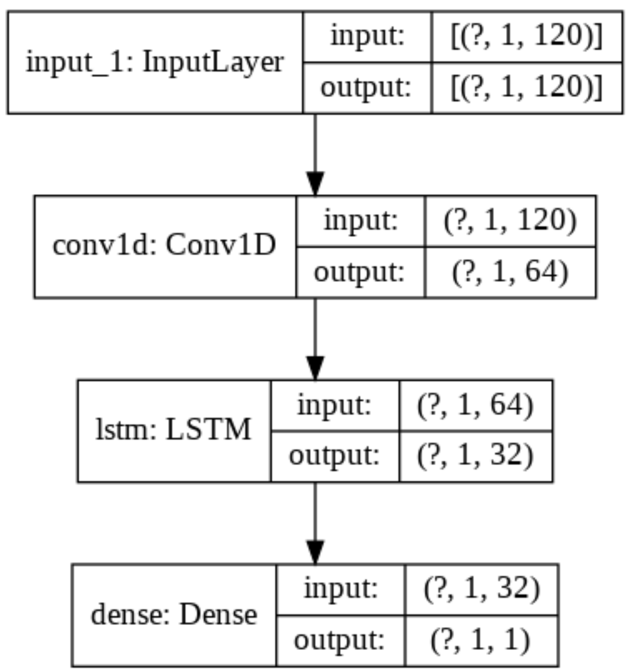

Figure 3-2: Developed architecture as a sequential model using a convolutional layer, dense layers, CNN, and LSTM

7- An SVR method is applied to the Apple dataset with a lookback window of 30 days. Here, all daily stock variables are considered as inputs of the model (i.e., Open price, Close price, Daily high, Daily low, Adjusted close price, and Volume of trades) and therefore create 180 total features.

8- Model is trained on 75% of the dataset, and 1-day prediction is performed.

9- Grid Search method is used to find the optimal parameters of the model.

10- Steps 8, 9, and 10 are carried out for the other three stocks.

11- Another model on the entire Apple dataset is trained using SVR and is then used to predict the other three stock prices. This is to investigate whether it is possible to use an SVR model trained on one stock to make forecasts on another one.

4 Data Analysis

4.1 Method 1: LSTM

In this Section, the data analysis based on LSTM approach is discussed.

4.1.1 Stock price prediction using LSTM

Convergence rate and one-day forecasts of the Adjusted close price as the response variable is presented for AAPL. Similarly, LSTM models are trained on MAST, FORD, and EXON datasets, and predictions are presented in Appendix B. Mean Absolute Percentage Error (MAPE) for each case was calculated to evaluate the performance.

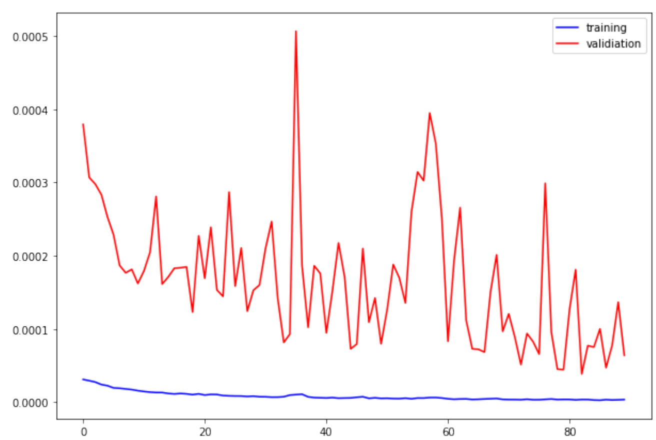

Figure 4-1: Convergence rate of LSTM for AAPL dataset

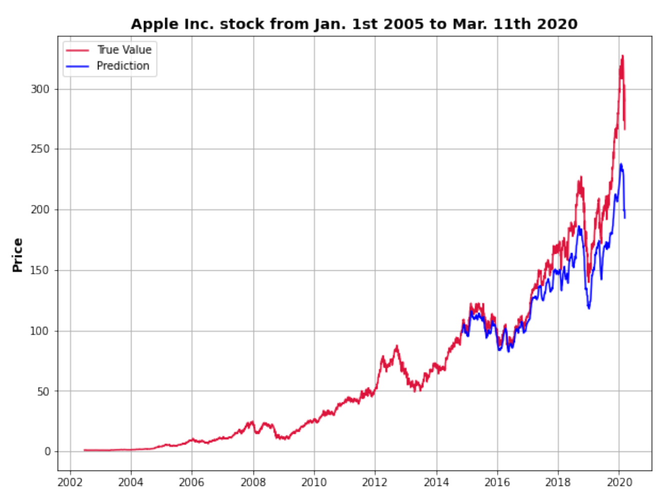

Figure **4-2**: Stock price predictions using LSTM for AAPL

4.1.2 One single LSTM model for all datasets

As explained earlier, we would like to assess the accuracy of a model trained on one stock dataset when it is used to forecast other stock prices. This is to investigate whether it is possible to use a model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using LSTM and is then used to predict the other three stock prices. As it is observed in the figures presented below, a model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

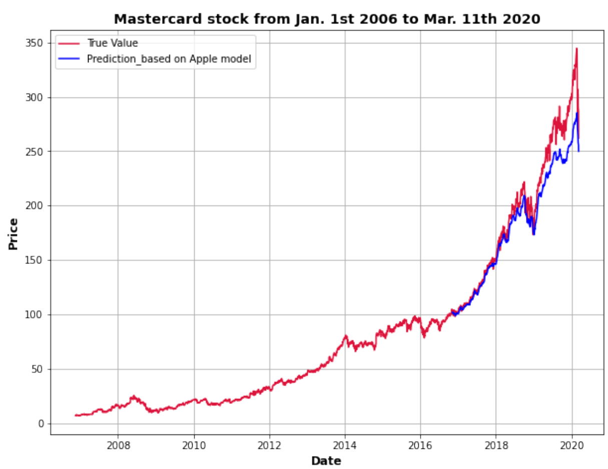

Figure **4-3**: MAST stock price predictions using LSTM trained on AAPL

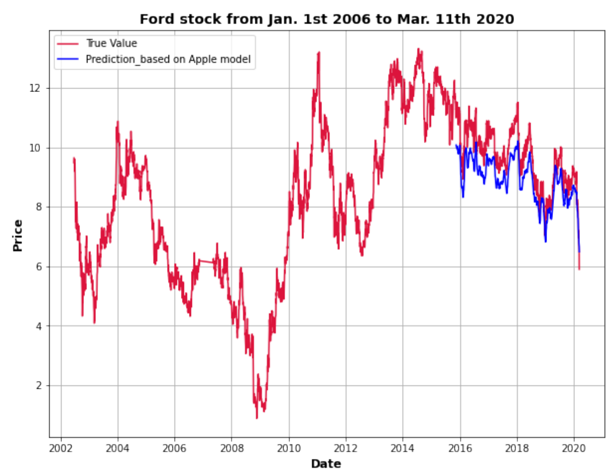

Figure **4-4**: FORD stock price predictions using LSTM trained on AAPL

Figure **4-5**: EXON stock price predictions using LSTM trained on AAPL

4.2 Method 2: CNN-LSTM

In this Section, results obtained from CNN-LSTM approach are discussed.

4.2.1 Stock price prediction using CNN-LSTM

Convergence rate and one-day forecast of the Adjusted close price as the response variable are presented for AAPL. Similarly, CNN-LSTM models are trained on MAST, FORD, and EXON datasets, and predictions are presented in Appendix B. MAPE for each case was calculated to evaluate the performance.

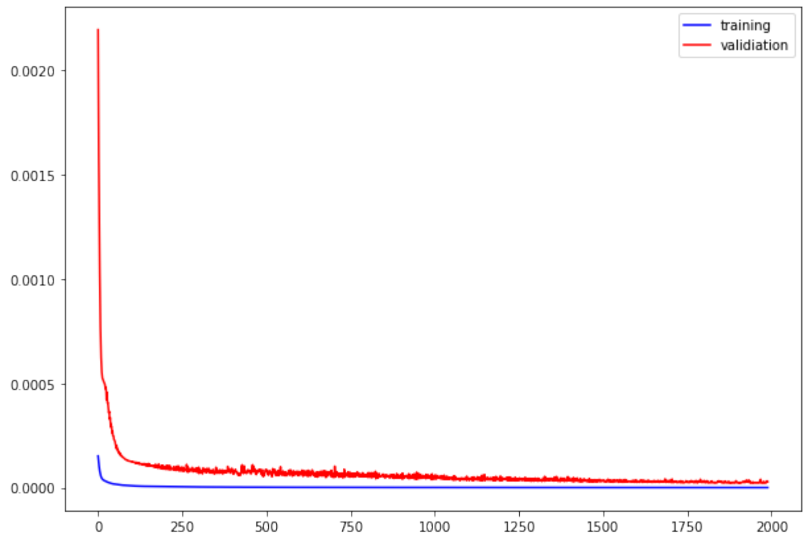

Figure **4-6**: Convergence rate of CNN-LSTM for AAPL dataset

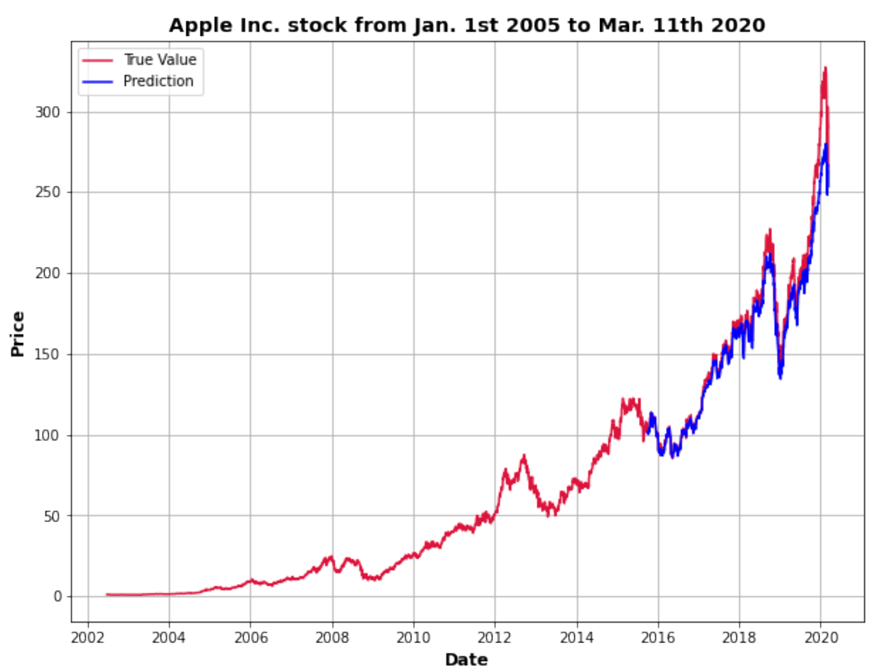

Figure **4-7**: Stock price predictions using CNN-LSTM for AAPL

4.2.2 One single CNN-LSTM model for all datasets

Similar to the previous Section, we would like to assess the accuracy of a model trained on one stock dataset when it is used to forecast other stock prices. This is to investigate whether it is possible to use a CNN model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using CNN and is then used to predict the other three stock prices. As it is observed in the figures presented below, a model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

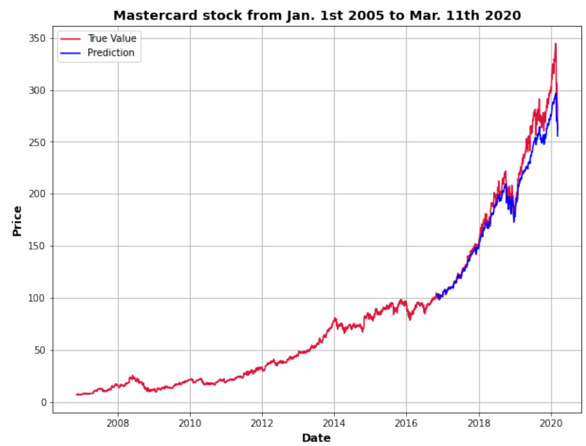

Figure **4-8**: MAST stock price predictions using CNN-LSTM trained on AAPL

Figure **4-9**: FORD stock price predictions using CNN-LSTM trained on AAPL

Figure **4-10**: EXON stock price predictions using CNN-LSTM trained on AAPL

4.3 Method 3: SVR

An SVR method is applied to each stock dataset with a lookback window of 30 days. Here, all daily stock variables are considered as inputs of the model (i.e., Open price, Close price, Daily high, Daily low, Adjusted close price and Volume of trades) and therefore create 180 total features. The model is trained on 75% of each dataset, with the kernel function being set to RBF. Grid Search is then conducted to determine the best combination of the following hyperparameters: C: 500, 1000, 2000, Epsilon: 0.001, 0.1, 0.5, Gamma: 0.001, 0.1, 0.5.
It turns out that for all datasets, C = 2000, Epsilon = 0.001, and Gamma = 0.001 is the best combination for this SVR model. The following figures present forecasts using the SVR model. MAPE for each case was calculated to evaluate the performance.

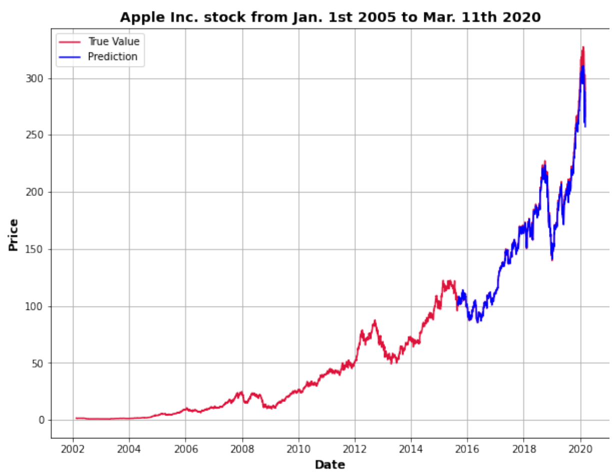

Figure **4-11**: Stock price predictions using SVR for AAPL

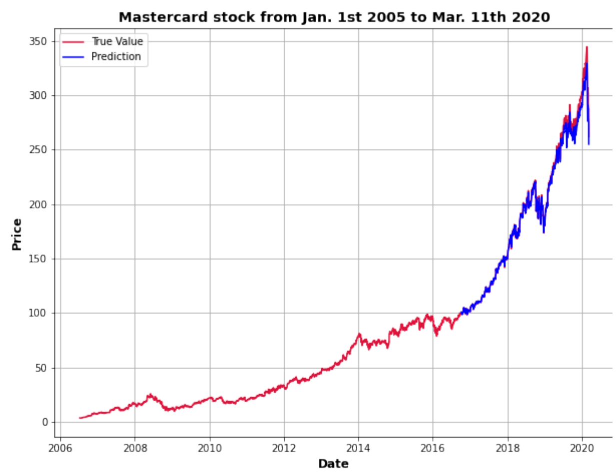

Figure **4-12**: Stock price predictions using SVR for MAST

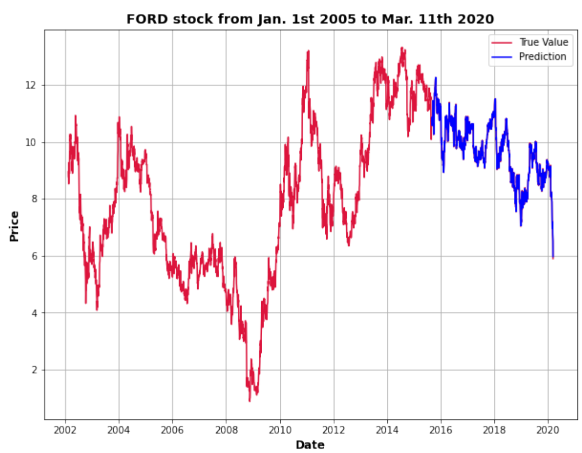

Figure **4-13**: Stock price predictions using SVR for FORD

Figure **4-14**: Stock price predictions using SVR for EXON

4.3.1 One single SVR model for all datasets

Similar to the previous Section, here we would like to assess the accuracy of the SVR model trained on one stock dataset when it is used to forecast other stock prices. This is to investigate whether it is possible to use an SVR model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using SVR and is then used to predict the other three stock prices. As it is observed in the figures presented below, an SVR model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

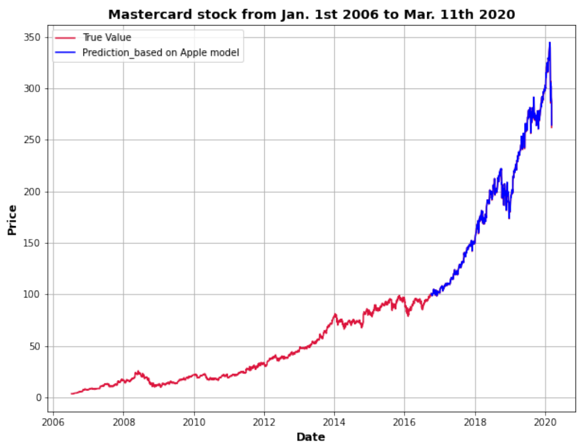

Figure **4-15**: MAST stock price predictions using SVR trained on AAPL

Figure **4-16**: FORD stock price predictions using SVR trained on AAPL

Figure **4-17**: EXON stock price predictions using SVR trained on AAPL

5 Discussions

From the figures provided in the Section 4, it is viewed that predictions are fairly accurate when the market is in a stable state (2015 – 2017 period), but as the volatility increases as in January 2020 forward, although the models still capture the general pattern, misalignments are observed in the forecast results and actual prices.

The following table summarizes the accuracies of different models in terms of the mean absolute percentage error (MAPE). This table proves that combining CNN and LSTM increases the accuracy compared to LSTM. Also, according to these results, the SVR model is able to predict stock prices with the highest accuracy compared to the other two methods. The reason is thought to be due to a more comprehensive set of features that were used in the SVR model.

It was also shown in this project that we can train a machine learning model on one stock dataset and use it to forecast different stock prices with a mild sacrifice in terms of accuracy. In the case of CNN model for predicting FORD stock price, for example, table below shows that a model trained on Ford dataset will perform with MAPE = 2.66 while a model trained on AAPL used to forecast FORD prices has MAPE of 8.68. In the Data Description Section, we showed that all these four stocks have positive correlations with each other, and this could be a reason why a model trained on one of them can predict the others.

Table 5-1: Summary of the performance of different methods

Model	Trained on Data from	Mean Absolute Percentage Error MAPE of each Machine Learning's prediction on			
		AAPL	MAST	FORD	EXON
LSTM	AAPL	13.75	6.67	8.68	9.71
LSTM	MAST	-	19.64	-	-
LSTM	FORD	-	-	2.66	-
LSTM	EXON	-	-	-	1.34
CNN	AAPL	2.18	4.78	6.77	7.39
CNN	MAST	-	4.17	-	-
CNN	FORD	-	-	0.55	-
CNN	EXON	-	-	-	0.77
SVR	AAPL	0.67	0.11	0.51	0.21
SVR	MAST	-	0.86	-	-
SVR	FORD	-	-	0.097	-
SVR	EXON	-	-	-	0.085

6 Future Work

This research was conducted while all group members had to work remotely due to unfavorable conditions caused by COVID-19 pandemic. Since the coordination of a group project was a challenge under such conditions, it is strongly recommended that further research be conducted in the future to address some shortcomings of this work. That is, using the same features for all models to improve comparability of results for SVR and CNN-LSTM models. Also, the effect of number of previous days on the future prices should be studied in further detail

References

[1] Sadia, K.H., Sharma, A., Paul, A., Padhi, S, Sanyal, S., (2019) Stock Market Prediction Using Machine Learning Algorithms, International Journal of Engineering and Advanced Technology, Volume 8, Issue 4.

[2] Box, G., Jenkins, G., (1970) Time Series Analysis: Forecasting and Control, Holden-Day.

[3] Coupelon, O., Neural network modeling for stock movement prediction, A state of the art, Blaise Pascal University.

[4] Huang, Y., (2019) Machine Learning for Stock Prediction Based on Fundamental Analysis, The University of Western Ontario.

[5] Huang, C., Huang, L., Han, T., (2012) Financial time series forecasting based on wavelet kernel support vector machine, IEEE Eighth International Conference on Natural Computation (ICNC).

[6] Wu, J., Lu, C., (2012) Computational Intelligence Approaches for Stock Price Forecasting, IEEE International Symposium on Computer, Consumer and Control (IS3C), pp. 52 – 55.

[7] Kao, L., Chiu, C., Lu, C., Chang, C., (2013) A hybrid approach by integrating wavelet-based feature extraction with MARS and SVR for stock index forecasting, Decision Support Systems, Volume 54, Issue 3, pp. 1228 – 1244.

[8] Tipirisetty, A., (2018) Stock Price Prediction using Deep Learning, San Jose State University.

[9] Cortes, C., Vapnik, V., (1995) Support-vector networks, Machine Learning 20 (3), pp. 273–297.

[10] https://towardsdatascience.com/support-vector-machines-for-classification-fc7c1565e3

[11] Boser, B. E., Guyon, I. M., Vapnik, V. N., (1992) A Training Algorithm for Optimal Margin Classifiers, Proceedings of the Fifth Annual Workshop on Computational Learning Theory, ACM Press, Pittsburgh, PA, pp. 144–152.

[12] Vapnik, V. N., (1995) The Nature of Statistical Learning Theory, Springer, New York.

[13] Hochreiter, S., Schmidhuber, J., (1997) Long Short-Term Memory, Neural Computation 9 (8), pp. 1735–1780.

[14] https://ai.stackexchange.com/questions/6961/structure-of-lstm-rnns

[15] Gunduz, H., Yaslan, Y., Cataltepe, Z., (2017) Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations, Knowledge-Based Systems 137, pp. 138–148.

[16] Chong, E., Han, C., Park, F. C., (2017) Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies, Expert Systems with Applications 83, pp. 187–205.

[17] Chen, K., Zhou, Y., Dai, F., (2015) A LSTM-based method for stock returns prediction: A case study of China stock market, IEEE International Conference on Big Data (Big Data), Santa Clara, CA, pp. 2823–2824.

Appendix A

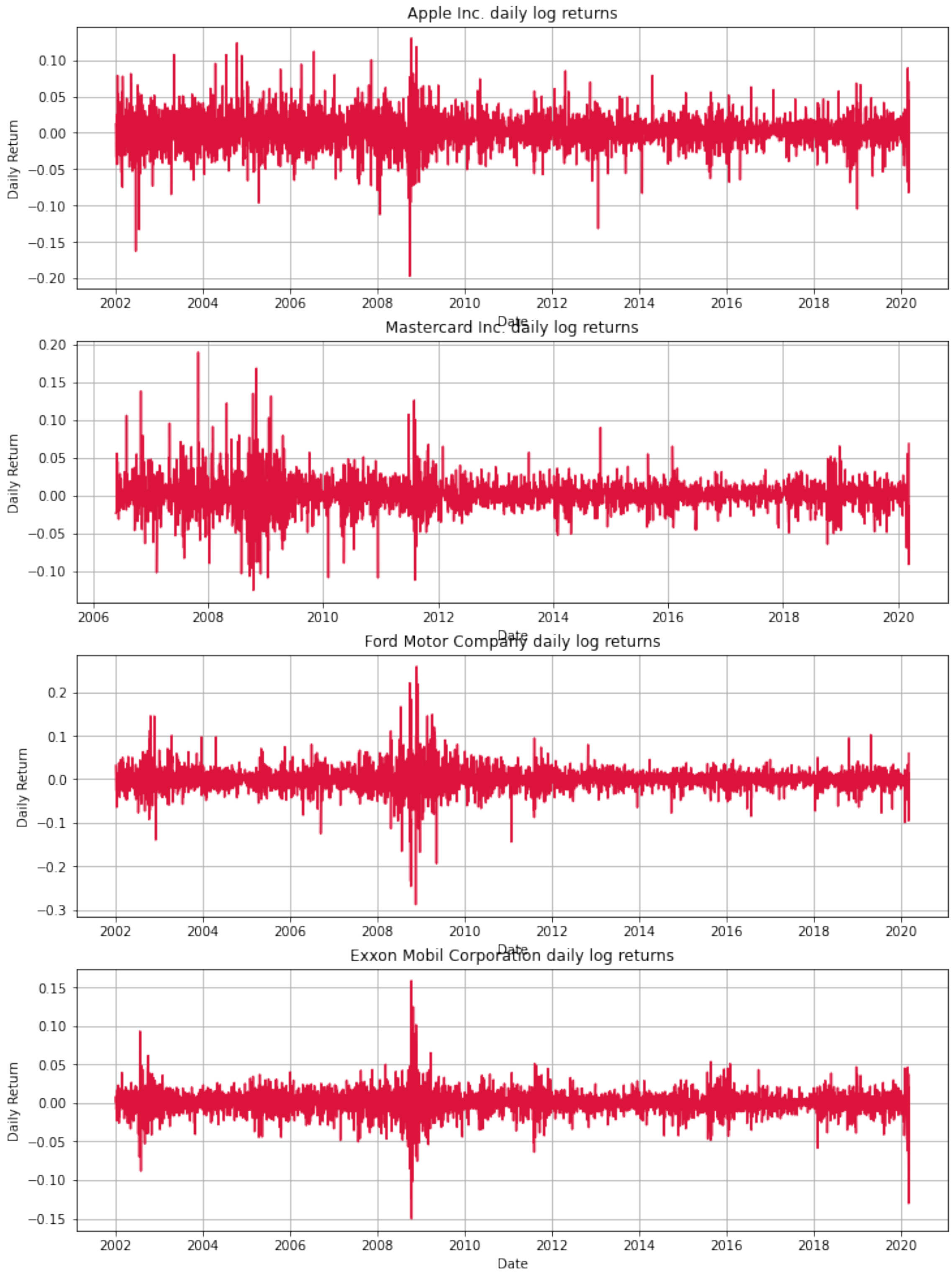

Figure **A-1**: Daily log-returns for Apple, MasterCard, Ford, and ExxonMobil

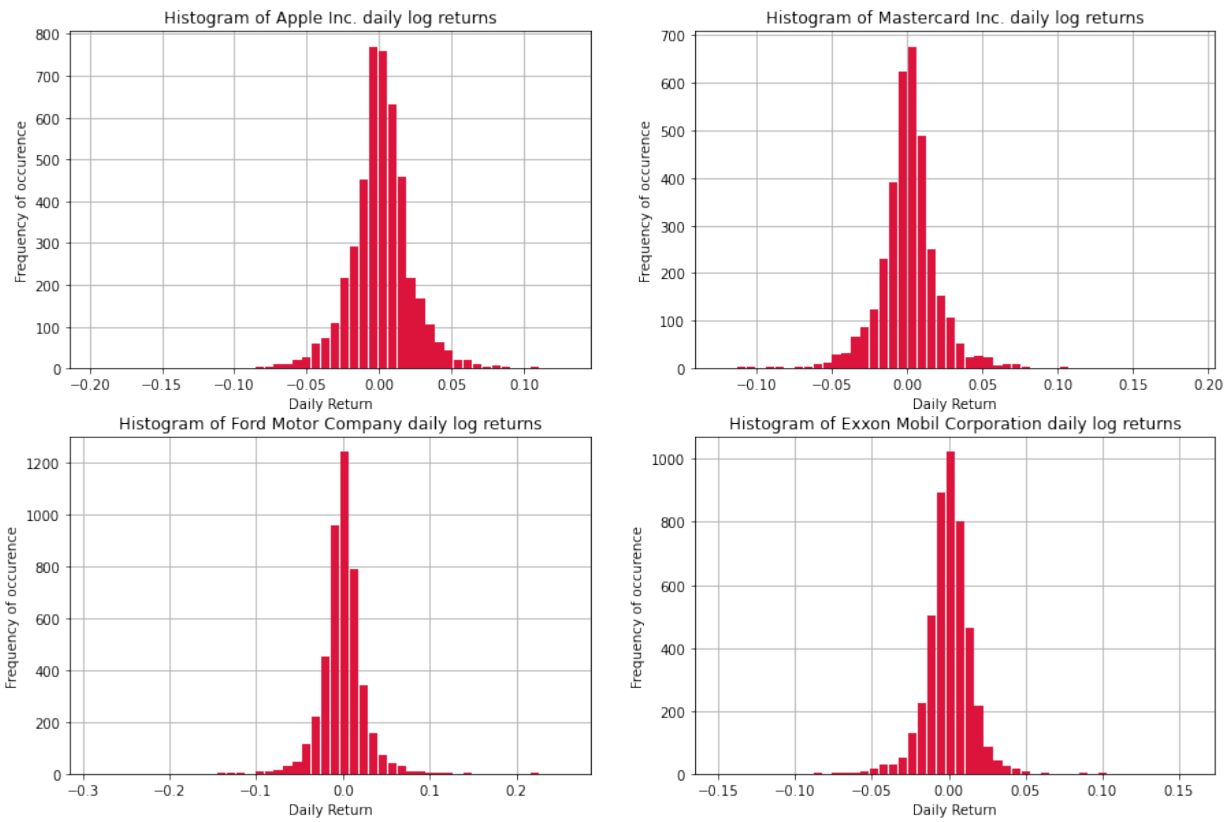

Figure **A-2**: Histograms of daily log-returns for Apple, MasterCard, Ford, and ExxonMobil

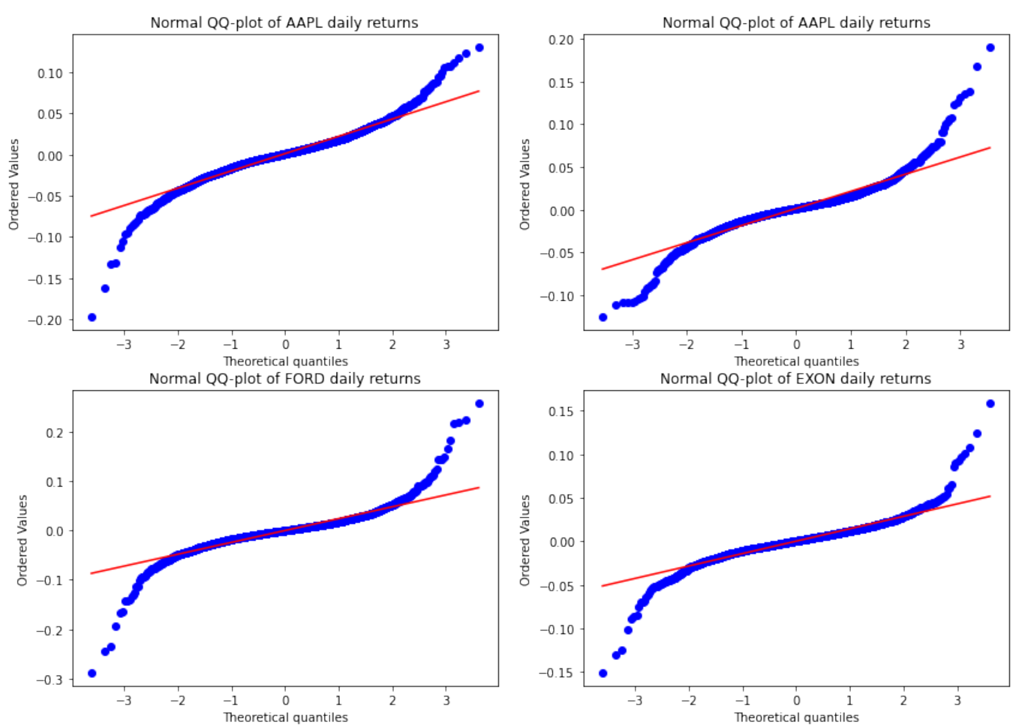

Figure **A-3**: Comparison of daily log returns distribution to Normal distribution

.

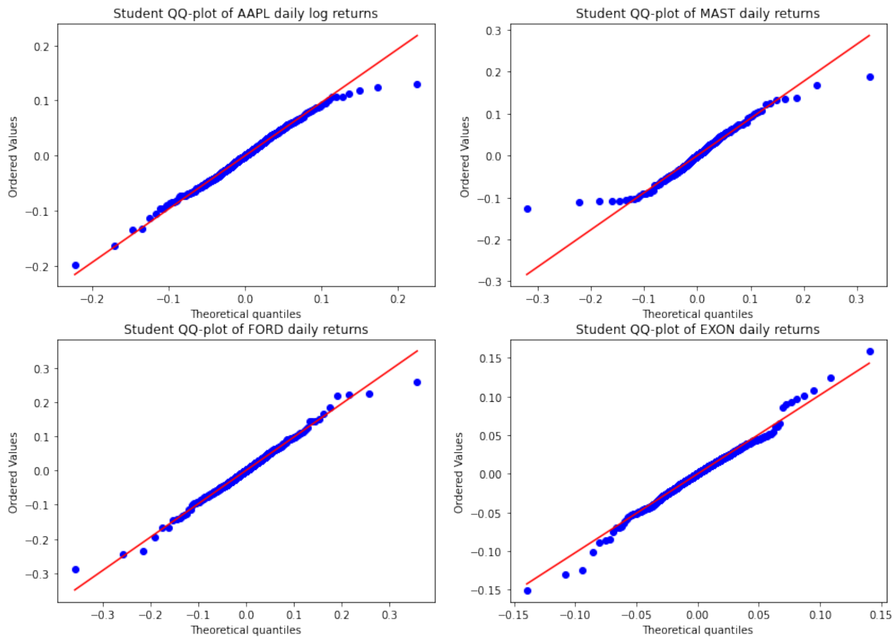

Figure **A-4**: Comparison of daily log returns distribution to Student's t-distribution

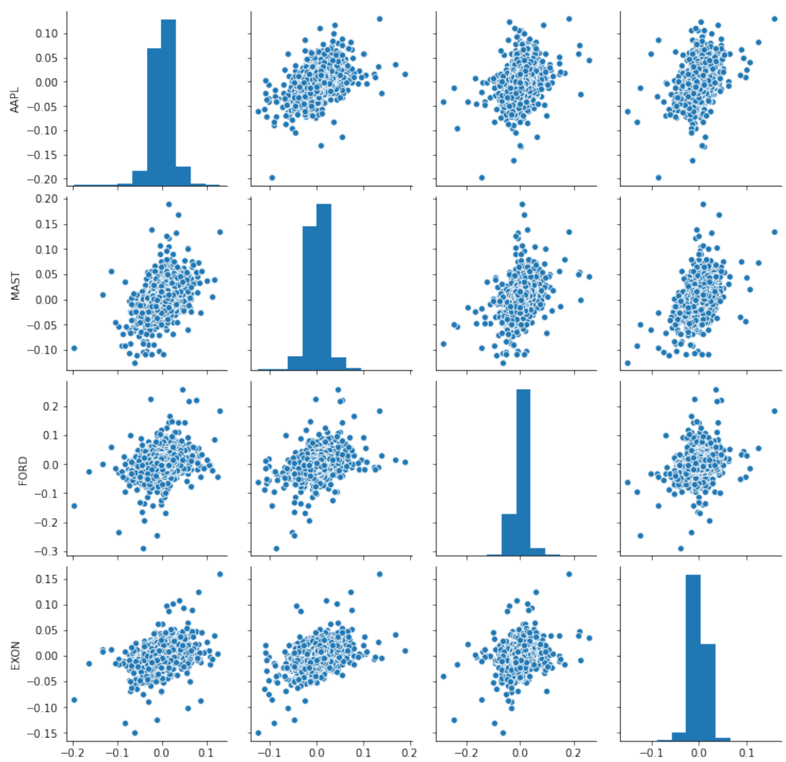

Figure **A-5**: Pair-plots of log-returns for Apple, MasterCard, Ford, and ExxonMobil

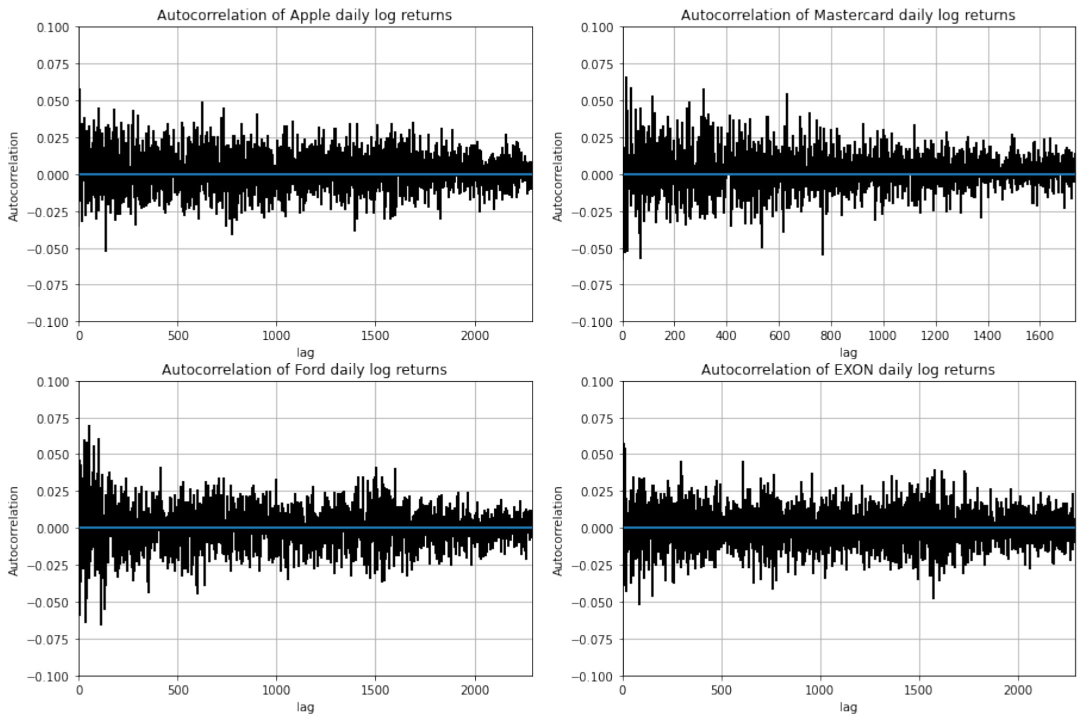

Figure **A-6**: Autocorrelation of log-returns for Apple, MasterCard, Ford, and ExxonMobil

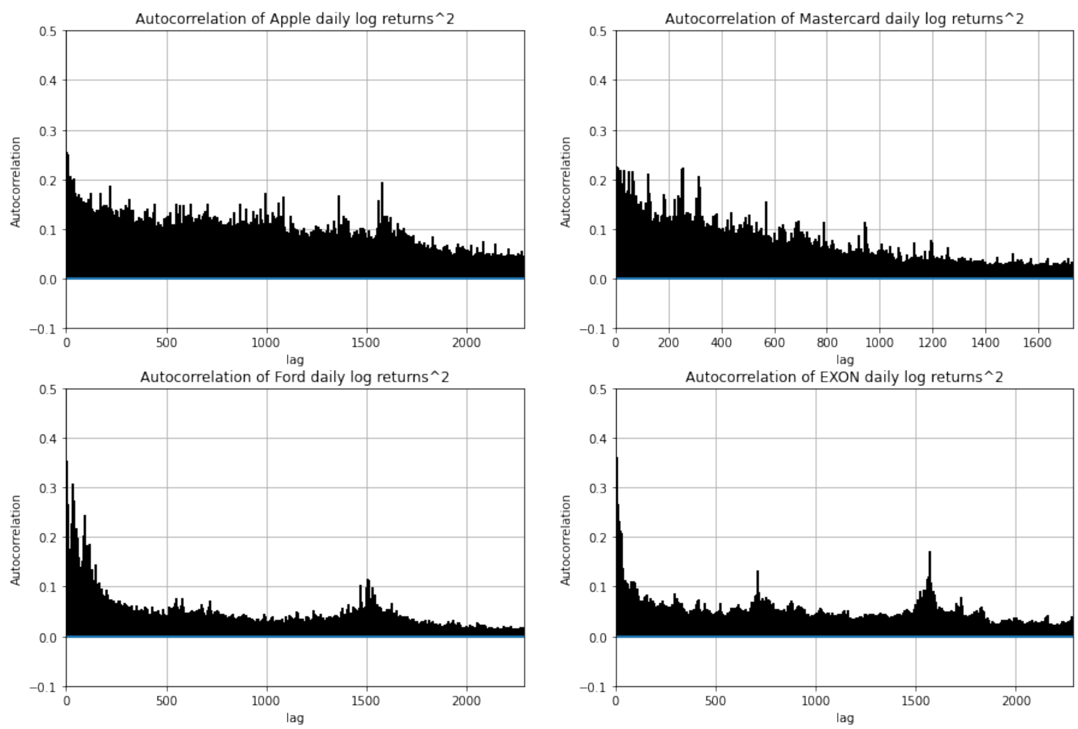

Figure **A-7**: Autocorrelation of squared of log-returns for Apple, MasterCard, Ford, and ExxonMobil

Appendix B

Figures below show the convergence rates and prediction results of different models trained to forecast MAST, FORD, and EXON prices.

B-1 MAST stock price prediction using LSTM

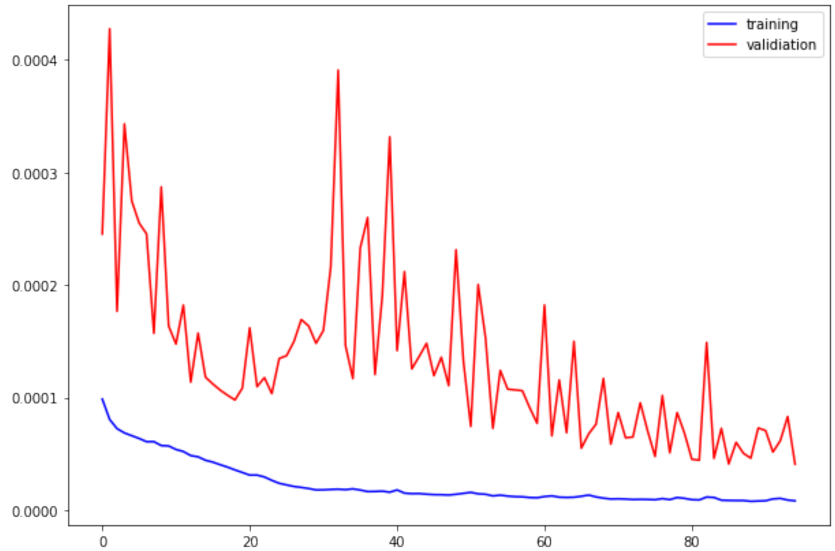

Figure **B-1**: Convergence rate of LSTM for MAST dataset

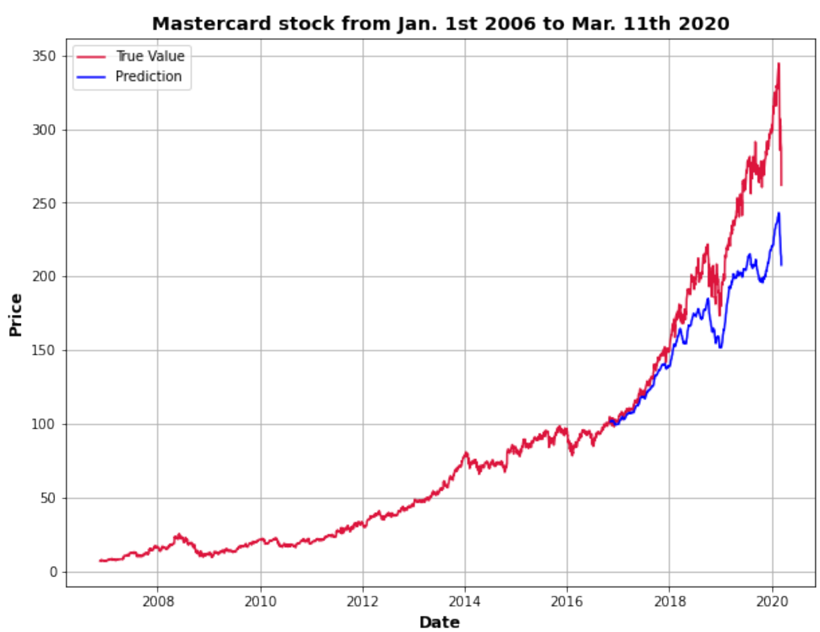

Figure **B-2**: Stock price predictions using LSTM for MAST

B-2 FORD stock price prediction using LSTM

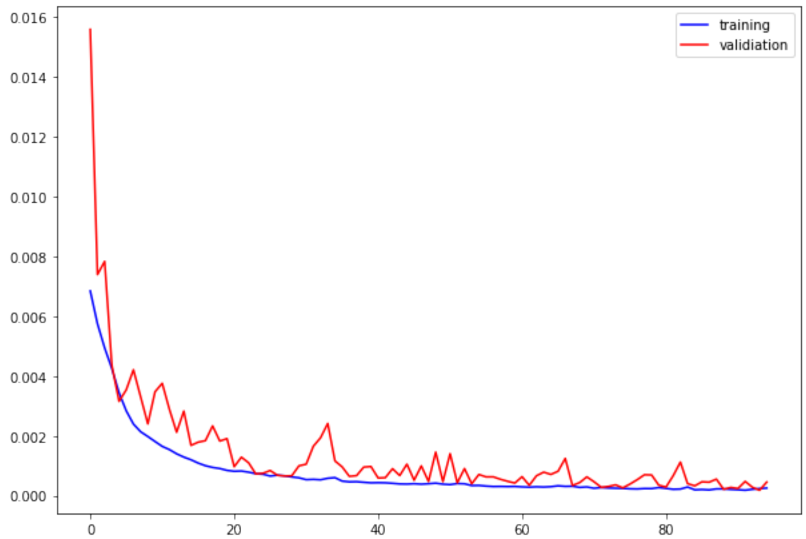

Figure **B-3**: Convergence rate of LSTM for FORD dataset

Figure **B-4**: Stock price predictions using LSTM for FORD

B-3 EXON stock price prediction using LSTM

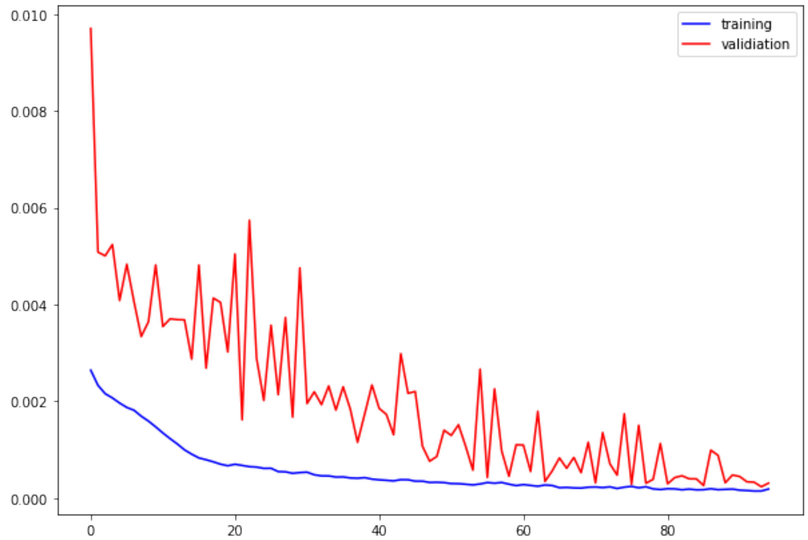

Figure **B-5**: Convergence rate of LSTM for EXON dataset

Figure **B-6**: Stock price predictions using LSTM for EXON

B-4 MAST stock price prediction using CNN-LSTM

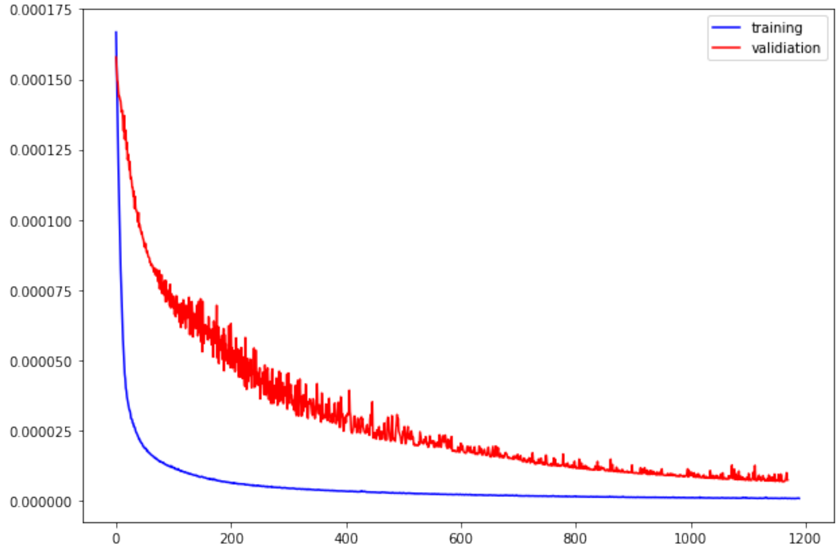

Figure **B-7**: Convergence rate of CNN-LSTM for MAST dataset

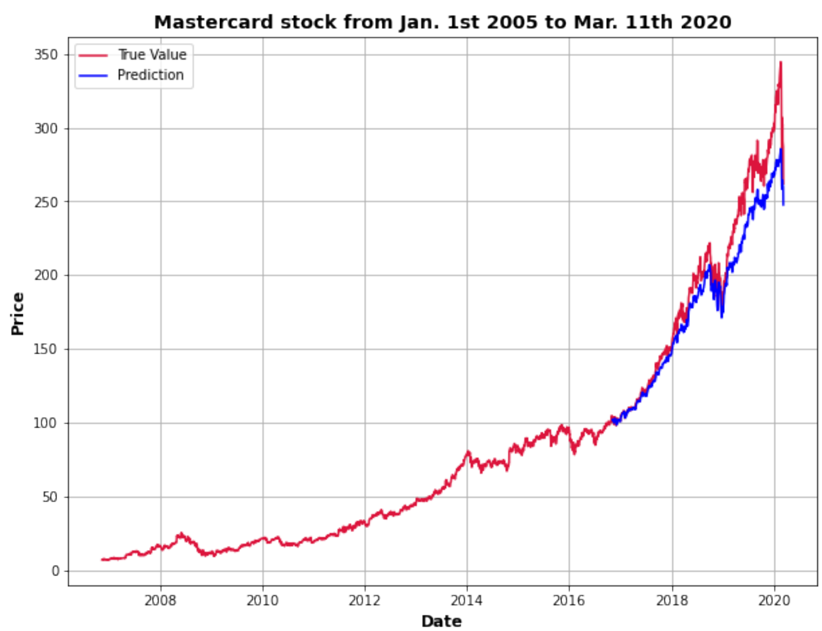

Figure **B-8**: Stock price predictions using CNN-LSTM for MAST

B-5 FORD stock price prediction using CNN-LSTM

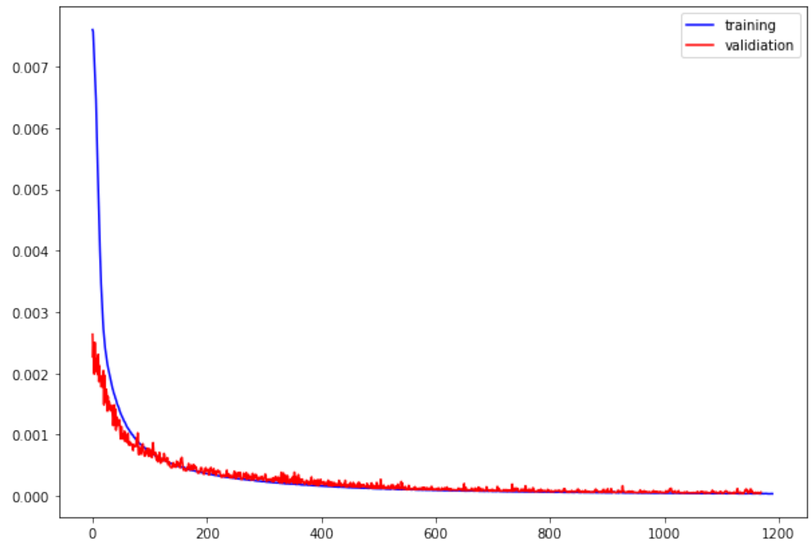

Figure **B-9**: Convergence rate of CNN-LSTM for FORD dataset

Figure **B-10**: Stock price predictions using CNN-LSTM for FORD

B-6 EXON stock price prediction using CNN-LSTM

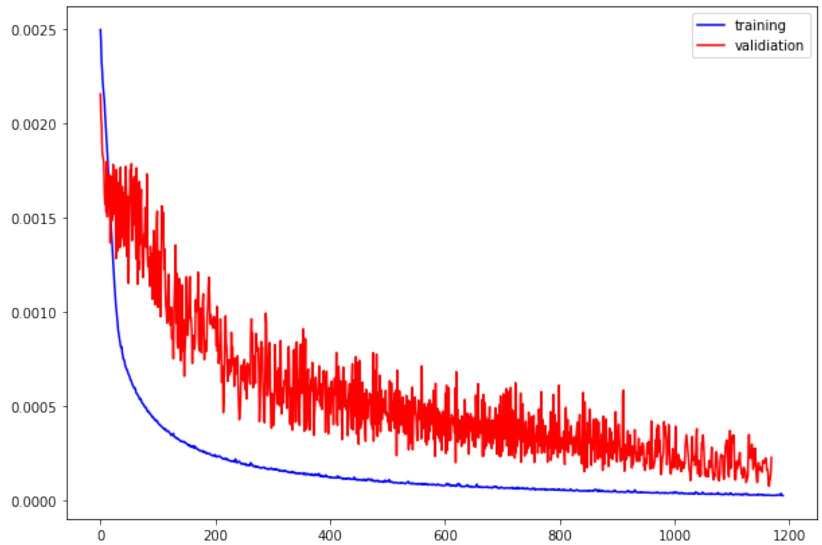

Figure **B-11**: Convergence rate of CNN-LSTM for EXON dataset

Figure **B-12**: Stock price predictions using CNN-LSTM